

# Implementing the Minimum-Misclassification-Error Energy Function for Target Recognition

Brian A. Telfer and Harold H. Szu  
Naval Surface Warfare Center, Code R44  
Silver Spring, MD 20903

## Abstract

Several new energy functions are proposed and tested for synthesizing neural networks for pattern recognition. Given a training set that adequately represents the actual class distributions, it is commonly desired that a neural network minimize the number of misclassified training vectors. However, the ubiquitous sigmoid-Least-Mean-Squares ( $\sigma$ -LMS) energy function does not generally produce such a network. Therefore, we have advanced a new Minimum-Misclassification-Error (MME) energy function to achieve this. We propose and test three new but related energy functions to achieve different classification goals. First is a minimum-cost function, which allows different costs for misclassifications from different classes. Second is a Neyman-Pearson function, which minimizes the number of misclassifications for one class given a fixed misclassification rate for the other class (i.e., fixed false alarm rate). Last is a minimax function, which minimizes the maximum number of misclassifications when the *a priori* probabilities of each class are unknown. Unlike their classical classifier counterparts, these energy functions operate directly on a training set, and do not require that class probability distributions be known.

## 1 Introduction

For automatic target recognition or other types of pattern recognition, we want a classifier that minimizes the probability of misclassifying test data. When the network weights are computed, the distributions of test data are known only through the training set. Therefore, given a training set that adequately represents the underlying class distributions, we want a network that minimizes the number of misclassified training samples. It has been shown [1] that LMS (either  $\sigma$ -LMS as normally used in backpropagation [2] or linear-LMS as used in the Widrow-Hoff algorithm [3]) does not produce a minimum-misclassification-error (MME) solution. Minimizing the misclassification error is vital in military applications, since a misclassification can produce a mission failure (for cruise missiles, ship defense, etc.).

We have proposed [4] an MME neural network to minimize the misclassification error. Others have also recognized that this is a desirable goal [5-7]. We demonstrate the MME energy function together with three new extensions: a minimum-cost classifier (which allows different costs for misclassifications from different classes), a Neyman-Pearson classifier (which minimizes the misclassification for one class for a given misclassification rate for the other class), and a minimax classifier (which minimizes the maximum misclassification rate possible when the *a priori* probabilities are unknown). The namesakes of these classifiers [8] require that the class distributions be known, but these new classifiers operate directly on a training set.

The neural network implementation of these classifiers is more powerful than the classical paradigm because the class density functions do not need to be known *a priori*. These neural network implementations also do not require that the densities be estimated from the training set before determining class boundaries; the neural network computes optimal class boundaries directly from the training set. Also, a neural network implementation allows massively parallel computing for real-time learning and recall.

Section 2 formulates all of the classifiers. Section 3 details the MME energy function and provides examples. The three new classifiers are demonstrated in Section 4. Section 5 provides comments on implementation and Section 6 a conclusion.

## 2 Formulation

Following notation from [8,9], let  $\omega_i$  denote the  $i$ -th class,  $P(\omega_i)$  be the *a priori* probability of that class, and  $p(x|\omega_i)$  be the probability density for a 1-D feature measurement  $x$  from  $\omega_i$ . In a two-class problem, there are two types of errors, those in which samples from  $\omega_1$  are misclassified and those in which samples

from  $\omega_2$  are misclassified. These can be written as

$$\varepsilon_1 = \int_{\Omega_2} p(x|\omega_1)dx, \quad \varepsilon_2 = \int_{\Omega_1} p(x|\omega_2)dx, \quad (1)$$

where  $\Omega_i$  is the region in which  $x$  is classified as  $\omega_i$ . Then the total error is

$$\varepsilon = P(\omega_1)\varepsilon_1 + P(\omega_2)\varepsilon_2. \quad (2)$$

For the case where we do not know the exact distributions and have only a training set, we approximate  $\varepsilon_1$ ,  $\varepsilon_2$ , and  $\varepsilon$  as

$$\hat{\varepsilon}_1 = \# \text{ misclassified class 1 training vectors} / N_1 \quad (3)$$

$$\hat{\varepsilon}_2 = \# \text{ misclassified class 2 training vectors} / N_2, \quad (4)$$

$$\hat{\varepsilon} = (N_1/N)\hat{\varepsilon}_1 + (N_2/N)\hat{\varepsilon}_2, \quad (5)$$

where  $N_1$  and  $N_2$  are the number of training vectors from  $\omega_1$  and  $\omega_2$  and  $N = N_1 + N_2$ . In the infinite sample case,  $\hat{\varepsilon}$  converges to  $\varepsilon$ . Eqs. 3-5 are a general definition for our MME energy function, for which a specific equation is given in Section 3. Minimizing  $\hat{\varepsilon}$  minimizes the number of misclassified training vectors.

Of course, minimizing the probability of error is not the only possible goal for a classifier. If the costs of misclassifying each class are different (say  $c_1$  and  $c_2$  for  $\omega_1$  and  $\omega_2$ ), then the goal is a minimum-cost classifier which should minimize

$$c_1\hat{\varepsilon}_1 + c_2\hat{\varepsilon}_2. \quad (6)$$

A Neyman-Pearson classifier minimizes  $\varepsilon_1$  for a fixed  $\varepsilon_2 = \varepsilon_0$  (i.e. fixed false alarm rate). We can write a energy function for this as

$$\hat{\varepsilon}_1 + c(\hat{\varepsilon}_2 - \varepsilon_0)^2, \quad (7)$$

where  $c$  is a weighting coefficient. Note that the second term becomes zero when  $\hat{\varepsilon}_2 = \varepsilon_0$ . Both Eqs. 6 and 7 can specify various points along the Receiver Operating Characteristic (ROC) curve, but in different ways. Eq. 6 is more desirable when the misclassifications have different costs, and Eq. 7 is more desirable when a false alarm rate is specified.

The misclassification probability is a function of the *a priori* probabilities. In cases where these are not known or can vary, it is useful to minimize the maximum error that can occur for any set of *a priori* probabilities. This is called a minimax classifier. A derivation [8] shows that for the minimax classifier,  $\varepsilon_1 = \varepsilon_2$ . An energy function to minimize this is given by

$$\hat{\varepsilon}_1 + c(\hat{\varepsilon}_2 - \hat{\varepsilon}_1)^2. \quad (8)$$

Note that the second term becomes zero when  $\hat{\varepsilon}_1 = \hat{\varepsilon}_2$ . A straightforward extension (which we do not test in this paper) of the minimax classifier incorporates different misclassification costs for each class.

These types of classifiers already exist when the class densities are known, just as the Bayes classifier is the minimum-error classifier when the class densities are known. Our contribution is to extend all of these to the case of classifiers that are synthesized from a training set by minimizing an energy function.

### 3 Minimum-Misclassification-Error Energy Function

In order to define and demonstrate the MME energy function, we first introduce our notation. Let  $\mathbf{w}$  be a  $L \times 1$  weight vector that connects an input vector  $\mathbf{x}$  with a single output  $\text{sgn}(\mathbf{w}^T \mathbf{x})$ , where the superscript  $T$  denotes transposition and  $\text{sgn}(\cdot)$  denotes the signum function ( $\text{sgn}(z) = 1$  if  $z \geq 0$ ;  $\text{sgn}(z) = -1$  otherwise). For a two-class problem, a positive output indicates one class and a negative output indicates the other. A training set with  $N$  training vectors is denoted as  $\mathbf{x}^n$ ,  $n = 1 \dots N$ . Although we are demonstrating the MME energy function with a very simple classifier, the extension of the energy function to multilayer feedforward networks is straightforward.

The commonly used  $\sigma$ -LMS energy function is given for this simple classifier by

$$E_{\sigma-LMS} = \sum_{n=1}^N \{d^n - \sigma[\mathbf{w}^T \mathbf{x}^n]\}^2, \quad (9)$$

where  $d^n$  is the  $n$ -th desired output and  $\sigma$  is a sigmoidal function. A more natural energy function for pattern recognition simply counts the number of training vectors that the network misclassifies. This is given by Eqs. 3-5, with  $E_{MME} = \hat{\varepsilon}$ , and Eqs. 3 and 4 defined by

$$\hat{\varepsilon}_i = \frac{1}{N_i} [N_i - \sum_{n=1}^{N_i} \text{step}(d^n \mathbf{w}^T \mathbf{x}^n)] = \frac{1}{N_i} \sum_{n=1}^{N_i} [1 - \text{step}(d^n \mathbf{w}^T \mathbf{x}^n)], \quad (10)$$

where  $d^n$  is now the desired output sign ( $\pm 1$ ) and  $\text{step}(z) = 1$  if  $z \geq 0$ ;  $\text{step}(z) = 0$  otherwise. When the desired output sign is the same as that of the actual output  $\mathbf{w}^T \mathbf{x}^n$ ,  $\mathbf{x}^n$  is correctly classified, the step function simplifies to 1, and the count of misclassifications is reduced by one. When the desired output sign and actual output sign differ,  $\mathbf{x}^n$  is misclassified, the step function simplifies to 0, and the count of misclassifications is not reduced. It is interesting to note that Eq. 10 is an  $L1$  norm while Eq. 9 is an  $L2$  norm (just as the city-block distance is an  $L1$  norm while the Euclidean distance is an  $L2$  norm).

To perform gradient descent requires that the energy function be differentiable. Since a step function is not differentiable, we approximate it as a sigmoid function of variable steepness. As the sigmoid is steepened, it better approximates the step function and the energy function better approximates  $E_{MME}$ . This sigmoid is given as

$$\sigma_\tau(z) = \frac{1}{1 - \exp(-z/\tau)}, \quad (11)$$

where  $\tau$  is the steepening parameter. Note that  $\sigma_1(z)$  is the standard sigmoid function used in back-propagation, and that  $\sigma_0(z)$  is the step function in Eq. 10. Thus, the energy function we minimize with gradient descent is based on

$$\hat{\varepsilon}_i = \frac{1}{N_i} \sum_{n=1}^{N_i} [1 - \sigma_\tau(d^n \mathbf{w}^T \mathbf{x}^n)], \quad \tau \rightarrow 0. \quad (12)$$

To demonstrate the MME and  $\sigma$ -LMS classifiers, we use the training set shown in Figure 1 and a test set (not shown), each set having 100 training vectors per class. We adopt a technique described elsewhere [10] so that  $\mathbf{w}$  defines a hyperspherical (circular in 2D) boundary. A quasi-Newton method (BFGS) [11] is used to minimize  $E_{\sigma-LMS}$  and  $E_{MME}$ . In minimizing  $E_{MME}$ , we start with  $\tau = 1$  and decrease it. Full details of the method, along with other test results, are provided elsewhere [10]. The class boundary produced by  $E_{\sigma-LMS}$  (Figure 1a) is clearly worse than that produced by  $E_{MME}$  (Figure 1b).

## 4 Simulations

We demonstrate the minimum cost, Neyman-Pearson and minimax classifiers on a two-class, two-feature problem. For all classifiers except the minimax,  $P(\omega_1) = P(\omega_2) = 0.5$ . Each class is drawn from a Gaussian probability density with covariance matrix  $\Sigma = \mathbf{I}$ . The  $\omega_1$  (called target) density has a mean (2, 2) and standard deviations (1, 2), while  $\omega_2$  (called clutter) has mean (0, 0) and standard deviations (2, 1). One thousand training and one thousand test samples were generated for each class. Figure 2 shows the training set with a boundary computed using  $E_{MME}$ . The  $E_{MME}$  and  $E_{\sigma-LMS}$  classifiers misclassified 16.9% and 18.2% of the test set, respectively. The 1.3% difference is small, but is consistent with other test results (including Figure 1) where  $E_{MME}$  outperforms  $E_{\sigma-LMS}$ .

A ROC plots the probability of detection (PD)  $1 - \hat{\varepsilon}_1$  vs. the probability of false alarm (PFA)  $\hat{\varepsilon}_2$ . The MME classifier produces one point on the ROC curve, but other points may be more desirable. Figure 3 plots ROCs found by varying the output threshold of the MME classifier (a suboptimal method), and the minimum-cost and Neyman-Pearson classifiers (with  $c = 100$  in Eq. 7) synthesized for different cost or false alarm values. (These curves are for the test set.) The minimum cost and Neyman-Pearson ROCs are

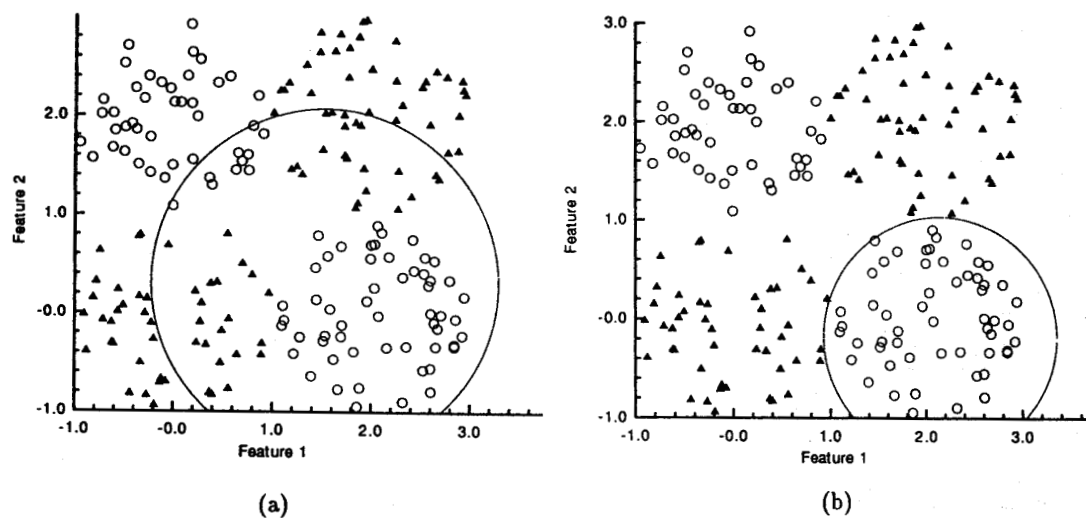


Figure 1: Boundary found by minimizing (a)  $E_{\sigma-LMS}$  and (b)  $E_{MME}$ .

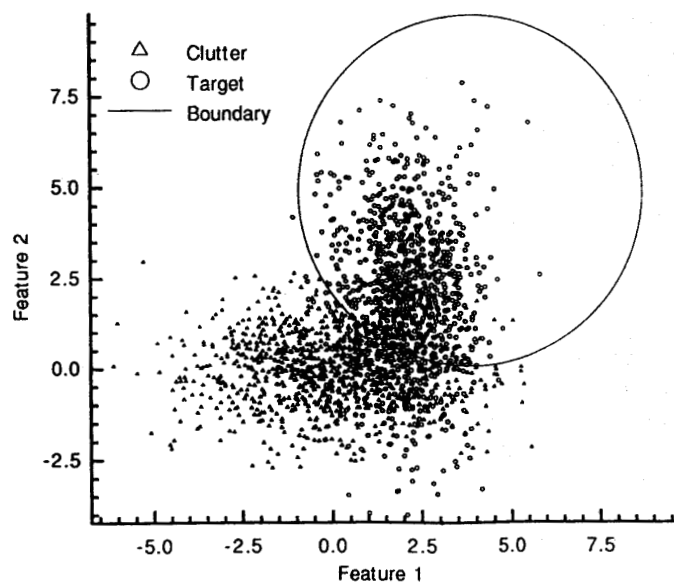


Figure 2: Training set and circular boundary found by MME.

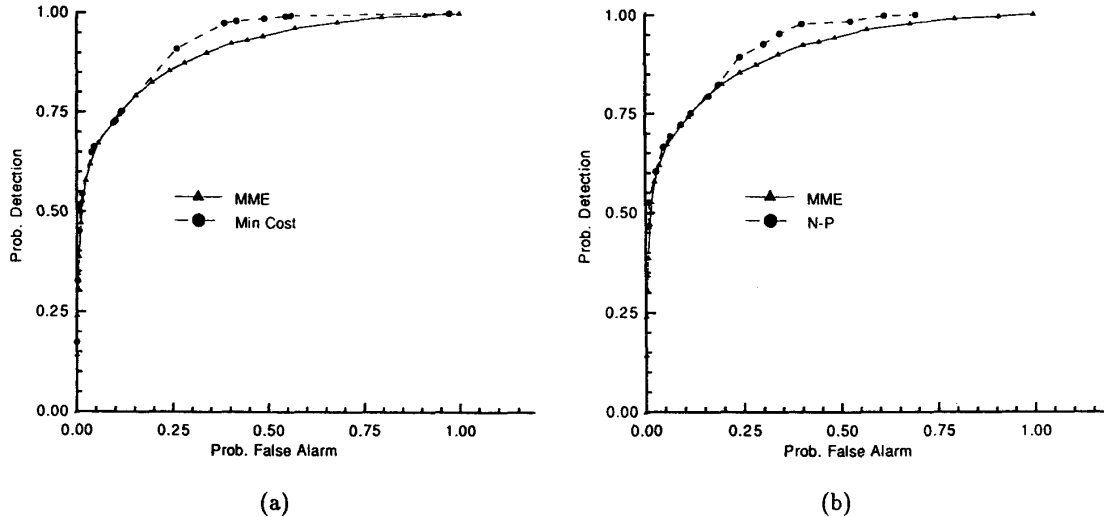


Figure 3: ROCs found by (a) minimum-cost and MME energy functions and (b) Neyman-Pearson and MME energy functions.

essentially identical. Either method may be preferable depending on whether the application is more easily characterized by different misclassification costs or by a desired false alarm rate. Note that for the Neyman-Pearson classifier, the plotted points occur at regular intervals that match the false alarm rates (e.g., 0.30, 0.35, 0.40, 0.50) specified in synthesis. Both methods are preferable to the suboptimal varying-threshold method. This is especially apparent above the “knee” of the curves, where the two methods perform up to 5.5% better than the suboptimal method. At the knee, all methods perform identically. This is because the operating point for the MME classifier is at the knee ( $PFA = 0.117$  and  $PD = 0.753$ ), and the other methods cannot perform any better at that or nearby points. Below the knee, the minimum-cost and Neyman-Pearson ROCs appear virtually identical to the MME ROC, but this is deceptive because of the very steep slopes of the curves. In fact, the minimum-cost and Neyman-Pearson classifiers outperform the MME below the knee by up to 6.5%-8% for a given false alarm rate. Specifically, for a false alarm rate of 0.007, for which all three classifiers generated an operating point, the test-set classification rate is 0.386, 0.451, and 0.465 for the MME, minimum-cost and Neyman-Pearson classifiers, respectively.

For the training and test sets defined above, the minimax classifier (with  $c = 200$  in Eq. 8) yields test set misclassification rates of  $\hat{\epsilon}_1 = 0.182$  and  $\hat{\epsilon}_1 = 0.183$ . Thus, the energy function operates as expected. For comparison, varying the threshold on the MME classifier so that  $\hat{\epsilon}_1 = \hat{\epsilon}_2$  for the training set yields  $\hat{\epsilon}_1 = 0.185$  and  $\hat{\epsilon}_1 = 0.182$  on the test set. Thus, by chance, the suboptimal approach produced essentially identical results. In general, the minimax classifier should perform as well or better than the suboptimal approach of varying the threshold of the MME classifier.

## 5 Implementation Comments

The feedforward operation (i.e., recall or testing) for the classifiers described in this paper is identical to that for the  $\sigma$ -LMS classifier, so identical hardware can be used. The learning phase obviously differs. Figure 4 gives block diagrams for implementing  $E_{\sigma-LMS}$  and  $E_{MME}$  in batch mode. Note that  $E_{\sigma-LMS}$  (Eq. 9) involves a subtraction of the desired output, while  $E_{MME}$  (Eq. 12) involves  $\sigma_\tau$ ,  $\tau \rightarrow 0$  (Eq. 11) and a multiplication by the desired sign, denoted by an encircled  $d^n$ .

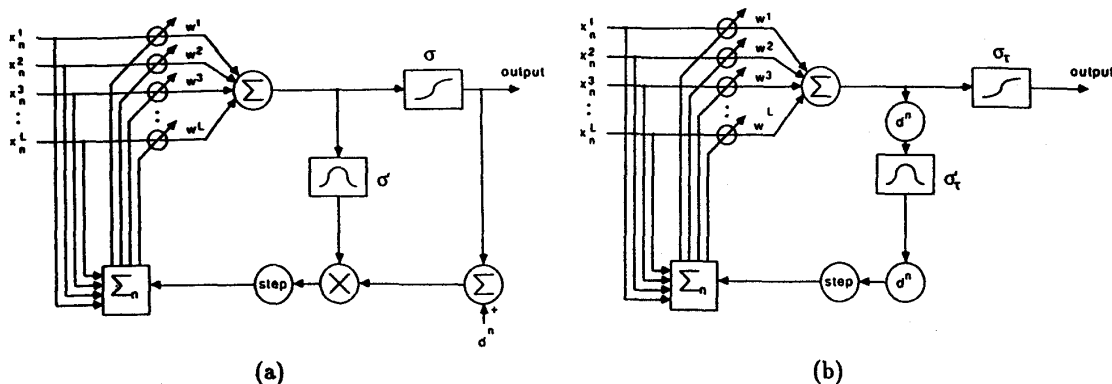


Figure 4: Implementations of (a)  $\sigma$ -LMS (modelled after [12]) and (b) MME learning.

## 6 Conclusion

We have demonstrated through a simple example that the MME classifier can dramatically outperform the  $\sigma$ -LMS classifier. This is true because  $E_{\sigma-LMS}$  does not minimize the training set misclassification rate (as others have also shown), while  $E_{MME}$  does. Building on  $E_{MME}$ , we have proposed three new energy functions that are useful for classification goals other than simply minimizing the misclassification rate. The minimum-cost energy function allows for different classes to have different misclassification costs. The Neyman-Pearson energy function allows the misclassification rate for one class to be minimized when the misclassification rate of the other class is fixed. The minimax energy function allows the worst-case misclassification rate to be minimized when the class *a priori* probabilities are unknown or can vary. These new energy functions were demonstrated through a simple example and shown to perform as desired and to outperform the suboptimal procedure of computing the MME classifier and varying its threshold. The network we used for demonstrating the concept was very simple (a single weight vector producing a hyperspherical boundary), but it is straightforward to apply these new energy functions to a multilayer feedforward network. Thus, one of these new energy functions, selected according to the particular classification goal, should be employed rather than  $E_{\sigma-LMS}$ , as long as the training set adequately represents the test set and the best performance possible is required.

## Acknowledgement

The support of this research by the NSWC Focused Technology Program in Neural Networks and an Office of Naval Research Young Navy Scientist Award is gratefully acknowledged.

## References

- [1] E. Barnard and D. Casasent, *IEEE Trans. Syst. Man and Cybern.*, 19, pp. 1030-1041, Sept./Oct. 1989.
- [2] D. Rumelhart, J. McClelland *et al.*, *Parallel Distributed Processing*, (MIT Press, Cambridge, 1986).
- [3] B. Widrow and M.E. Hoff, *IRE Western Electric Show and Convention Record, Part 4*, pp. 96-104, 1960.
- [4] H. Szu and B. Telfer, *Proc. IJCNN*, vol. II, p. A-916, July 1991.
- [5] J. Sklansky and G. Wassel, *Pattern Classifiers and Trainable Machines*, (Springer-Verlag, NY, 1981).
- [6] J. Kangas, T. Kohonen, and J. Laaksonen, *IEEE Trans. Neural Networks*, 1, pp. 93-99, March 1990.
- [7] E. Barnard, *IEEE Trans. Neural Networks*, 2, pp. 322-325, March 1991.
- [8] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed., (Academic Press, San Diego, 1990).
- [9] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, (John Wiley, NY, 1973).
- [10] B. Telfer and H. Szu, submitted to *Neural Networks*.
- [11] R. Fletcher, *Practical Methods of Optimization*, (John Wiley, NY, 1987).
- [12] B. Widrow and M.A. Lehr, *Proc. IEEE*, 78, pp. 1415-1442, Sept. 1990.